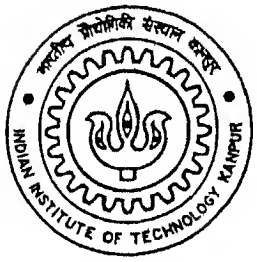
Design and Implementation of a Viterbi Decoder for a CDMA MODEM

by

Bupesh Pandita



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January 1998

Design and Implementation of a Viterbi Decoder for a CDMA MODEM

A Thesis Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Master of Technology

by
Bupesh Pandita

to the

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

January 1998

In
Pa
be

21.1.98

CERTIFICATE

This is to certify that the work contained in the thesis entitled **Design and Implementation of a Viterbi Decoder for a CDMA MODEM** by **Bupesh Pandita** has been carried out under my supervision and that this work has not been submitted elsewhere for a degree

2/2/98

Dr Subir K Roy
Department of Electrical Engineeering,
Indian Institute of Technology, Kanpur

Acknowledgements

I want to thank my thesis supervisor Dr Subir K Roy for introducing me to VLSI design I would also like to thank him for patiently listening to my arguments and also for the valuable suggestions that have gone into the making of this thesis I was given total freedom in pursuing this thesis and I am grateful to my thesis supervisor for this trust he reposed in me

I would also like extend my appreciation to Dr M U Siddiqui for pointing out certain subtleties which shaped the course of this thesis

I would like to thank Analog Devices for supporting my M Tech study through an Analog Devices fellowship

I also want to thank Mr Ashish Sirsao of SAS Pvt Ltd Bangalore for giving us access to the synthesis tool Pinnauq and also for the technical discussions we had over the email Last but not least I want to thank my so many friends for the interesting *tete-a-tete* in the Hall-5 canteen

Abstract

A major contribution of this thesis has been in the design and implementation of a Viterbi decoder. A Viterbi decoder is a very important block in any CDMA modem The aim was to design a 19 2 kbps ,256 state Viterbi decoder with added capability of catering to higher input data rates To the best of our knowledge none of the existing literature discusses Viterbi decoding implementation based on high level synthesis targetted for Field programmable Gate Arrays (FPGAs) This has been the focus in the present thesis

Besides the above, some of the issues such as organization of a path memory, decision memory, the decision memory reading techniques, and the clocking mechanism have been discussed We have also retained the bit-synchronization information, even though it made the normalization of the path metrics essential We have used very fast subtracters to implement normalization of the path metrics

In this thesis we explore an implementation methodolgy for rapid prototyping using FPGAs In designing the Viterbi decoder we have used some of the features discussed in the literature, however, we have attempted to compare their merits with other existing techniques and also the advantages that have accrued because of their use It has not been our attempt to give just another implementation of the Viterbi algorithm, but to create design guidelines which may be used in any future implementation of the decoder

Contents

List of Tables

List of Figures

Chapter 1

Design of a CDMA Modem – Some Implementation Issues

1.1 Introduction

In recent years there has been tremendous growth in a new technology for multiple access communication namely, code division multiple accessing (CDMA) This is in addition to the already existing technologies, such as frequency division multiple access (FDMA) and time division multiple accessing (TDMA) Though no technology is well suited for all situations, CDMA has some features which makes it attractive for multiuser communications In this chapter, we compare above three technologies with respect to their ability to address various issues such as capacity We also discuss some features which are special to the CDMA technology

1.2 Basics of CDMA

While TDMA or FDMA achieve user discrimination by time or frequency separability of the channels, CDMA assigns each user a signature sequence for separability In CDMA each signal consists of a different pseudorandom binary sequence that modulates the carrier, thereby spreading the spectrum of the waveform A large number of CDMA signals share the same frequency spectrum and also, there is no

orthogonality as created by time division When viewed in time or in frequency domain in CDMA we have signals on top of one another The signals are separated by a correlating receiver which collapses only the energy of the selected binary se quence The other user signals just represent interference to the pseudo random tuned receiver [1]

1 3 Some Different CDMA Systems

1 3 1 Direct Sequence Cellular CDMA (DS/CDMA)

In Direct Sequence Cellular CDMA (DS/CDMA) usually one frequency band is used for the base to mobile link (forward link) and a separate frequency band is used for mobile to base link (reverse link)

The two links forward and reverse differ with each other in some respects On the forward link a common pilot is used for channel estimation and time syn chronization The users can be orthogonalized The reverse link on the other hand does not enjoy these features It cannot be orthogonalized in view of the different locations as well as independent movements of the mobiles (Appendix A)

Characteristics of DS/CDMA

- **Universal Frequency Reuse** DS/CDMA cellular system can apply a uni versal one cell frequency reuse pattern If the traffic requirement at a certain location increases introduction of a new cell will be less restricted than in the case of either FDMA or TDMA

- **Power Control**

  1 **Reverse Link** The reverse link is designed to be asynchronous and hence is besieged with "near far"[1] problem A solution to the near far problem is the use of power control, which attempts to ensure that all

[1]near far problem exists when a transmitter of higher strength drowns another transmitter of lesser strength

signals from the mobiles within a given cell arrive at the cell site with equal power The power control required must be accuiate (typically within 1db) fast enough to compensate for Rayleigh fading and have a large dynamic range (80db) [2]

2 **Forward Link** The forward link does not suffer from the near far' problem Power control is applied to increase power to mobiles suffering from excessive intercell interference The requirements of dynamic range and speed are not as stringent as for the reverse link

- **Soft Handoff and Space Diversity** The users information is sent via two or more base stations which is diversity combined by the user s receiver Power control of the mobile is coordinated by the base station that receives the strongest signal The soft handoff [3] provides a 'male before break' handoff transition

- **Coding** Coding redundancy can be regarded as part of the spreading Code rates of 1/2 and 1/3 are typically used

- **Voice Activity** To reduce multiple access interference (MAI) transmission is stopped when voice or data ativity is absent

- **Antenna Gain** Fixed sectored antennas and phased arrays accomplish re duction of MAI, and hence increase the user capacity Typically a three sec tored antenna is used

Current DS designs

We will consider the IS 95 standard It employs BPSK data and QPSK spreading on the forward link , in conjunction with synchronous intracell transmission The chip rate [2] is 1 228 MHz and forward error correction is employed with a K=9, 1/2 convolutional code The use of spread pilot allows coherent detection to take place

[2]the chip rate is equivalent to the code generator clock rate or in frequency hopping systems the hop rate

On the reverse link 64 ary binary orthogonal signalling is used in conjunction with a k=9 1/3 convolution code

1 3 2 Frequency Hopping Cellular CDMA

Almost all Frequency Hopping (FH) systems are slow frequency hopping (SFH) systems (ie multiple bits are transmitted on each hop) (Appendix A)

SFH was proposed for cellular systems in the literature in [4 5] and the use of such systems has been discussed in [6] The main difference in the performance between DS and FH is that a FH system can be approximately orthogonalized thus limiting the user interference

The potential advantages of this type of FH are

- Less total interference
- No near far problem
- Better performance during jamming
- No need for guard bands at the fringes of the spectrum

FH In commercial applications

FH [7] has been widely used in most military communications systems for the twin reasons of fast synchronization and the absence of near far problem

In commercial applications for a DS system by making the codes shorter, synchronization time is not constrained as in the military communications Further synchronizations techniques not practical in military systems are possible in commercial applications

In a consumer application, with full duplex signals the probability of collision with FH system is twice as high as in a military system with greater channels and lesser number of half duplex users The quality of service provided by FH system is not as good as that provided by the DS system, and hence, DS systems have outperformed FH systems in commercial/consumer applications

1 4 Capacity of CDMA

As recently as 1985 [8] a straightforward comparison of the capacity of CDMA with that of TDMA or FDMA was in favor of TDMA and FDMA However unlike TDMA or FDMA capacity in CDMA is interference limited and any reduction in interference directly leads to a linear increase in capacity Some of the factors like voice activity and spatial isolation have rendered the capacity of CDMA at least double than that of FDMA or TDMA under similar assumptions for a mobile satellite application

Due to nonorthogonality of the CDMA waveforms though the capacity with respect to an isolated cell in CDMA is less than that with respect to either FDMA or TDMA it is in the multicellular environment that the comparisons are reversed

Unlike TDMA or FDMA DS spread waveforms can be used to reject multipath returns, or enhance overall performance by diversity combining multipath returns in a RAKE receiver[3]

Another consideration in favor of CDMA is the frequency reuse factor For FDMA or TDMA frequencies used are not used in adjacent cells For example for analog AMPS system [2], a frequency reuse of one in seven is employed However with CDMA a possibility of a frequency reuse of one in one exists (frequency reuse of two in three has been used in [1]

CDMA also provides a natural way to exploit the bursty nature of a source for added capacity Using a voice activity of 1/2, in principle the capacity of CDMA can be doubled

[3]In a RAKE receiver signals arriving via different path delays are recombined for correlation detection

1 5 CDMA Versus FDMA

The four major factors [9] that have reversed the comparison of capacity in favor of CDMA are –

- Voice activity
- Spatial discrimination provided by satellite multibeam antenna
- Cross polarization frequency reuse
- Discrimination between multiple satellites providing co coverage

The voice activity greatly reduces the self noise of the spread spectrum system and utilizes the satellite downlink power more efficiently

In FDMA, the voice activity factor does not increase the capacity when the system is bandwidth limited but only reduces the satellite downlink power when the satellite is in a power limited mode FDMA systems are unable to exploit voice activity factor to improve the capacity of bandwidth limited mobile to hub links because of the delays inherent with synchronous orbit satellite

The capacity of CDMA system is further improved by multibeam antennas which allow a degree of frequency reuse Though a FDMA sytem can increase capacity by employing such antennas the increase in capacity by frequency reuse is substantial in the case of CDMA Frequency reuse is greater for CDMA because the antennas can be spaced by B(say) degrees and the entire frequency band reused every B degrees With a large number of beams, CDMA can reuse the entire frequency band in each antenna beam This is a factor of three superior to that obtainable with FDMA

CDMA can use the entire frequency band by utilizing the opposite senses of cross polarization Polarization isolation cannot usually be exploited by a FDMA system employing small mobile antennas because such antennas provide only limited cross polarization, usually less than the necessary cochannel channel interference ratio (C/I) required by FDMA

A possibility for increasing the channel capacity in CDMA is by using multiple satellites and by the coherent combining of signals transmitted between a satellite and all satellites in view

This technique is not available for FDMA Also, FDMA system operating in the bandwidth limited regime cannot have increased capacity by having available increased downlink power Thus a FDMA system operating in the bandwidth limited mode does not benefit from additional satellites as a way of increasing capacity unless every mobile terminal is equipped with a costly directive antenna capable of providing sidelobe rejection to result in adequate C/I performance in the adjacent satellite

1 6 Trellis Coded And Convolutionally Coded Spread Spectrum Multi-Access

In [10] the results indicate that convolutional codes provide superior performance than utilizing a trellis code [11] applied to the data symbols It has been observed that gain of the convolution code improves as lower rate convolution codes are used Though in a band limited environment application of low rate codes would penalize the bandwidth, but for Spread Spectrum (SS) applications low rate convolutional codes can be applied without increasing the bandwidth or decreasing the processing gain [3] Advantage of low rate convolution codes over trellis codes is the greater distance property and it has been observed that additional coding gain accruing out of greater distance properties of lower rate convolution codes outweighs any other factor against it

1 7 Orthogonal Code Techniques For CDMA

In [12] two realistic (quasi) orthogonal techniques have been compared The comparison was done using a single user matched filter receivers

The two systems considered were –

1 Walsh Hadamard Functions Based Orthogonal DS/SS CDMA system

2 Cold Functions Based Quasi orthogonal DS/SS CDMA system

1 7 1 Walsh-Hadamard DS/SS system

This particular system has been patented by Qualcomm Inc A carrier is assigned a Walsh Hadmard sequence [13] (M=64) and then the data is divided into identical I and Q components These components are overlaid with long PN sequences [period L=32768] This overlay protects WH spread signal from possible asynchronous interference due to multipath distortion or adjacent cell interference It also aids in code acquisition process

1 7 2 Gold Functions based DS/SS system

This system was patented by the European Space Agency (ESA) The main difference from the previous system is that a single spreading sequence is used for code acquisition and users' orthogonalization A QPSK signaling with independent I and Q spreading by means of two different Gold sequences is employed The main advantage of this signaling technique is the reduced bandwidth for a given bit rate and spreading factor The small residual self-noise due to the partial orthogonality of the Gold Sequences [14] is negligible as far as bit error rate (BER) and overall system efficiency are concerned

It has been observed in [12] that there is no clear winner in this comparison A minimum advantage of the WH functions based system in frequency selective fading channel is somewhat counterbalanced by decreased sensitivity of Gold functions based system to implementation losses and nonlinear distortions No substantial

difference is observed as far as overall bandwidth efficiency and carrier phase offset sensitivity are concerned Further it has been suggested that system level issues should be taken into considerations for an in depth comparison For instance initial code acquisition is presumably slower in WH functions based system but other high level functions such as soft handover or diversity reception may be easier in the WH functions based system

Chapter 2

CDMA Mobile Station Modem ASIC

2 1 Introduction

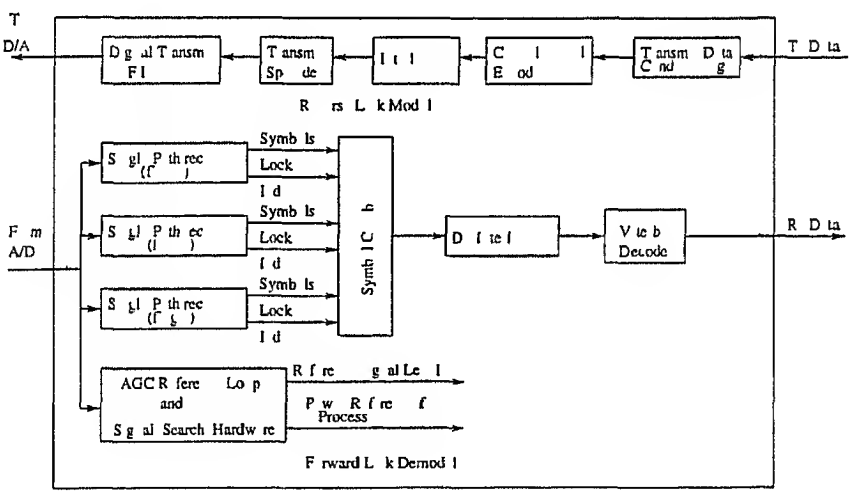


Figure 2 1 CDMA MODEM block diagram

Success of any satellite based communication system depends upon, low cost, small-size and low power consumption user terminals This basic requirement calls for state of the-art modem architecture, design and implementation With the advent of portable cellular telephones a need to fit the maximum amount of

circuitry in one integrated circuit has been felt The continual improvement of application specific integrated circuits (ASICs) makes a single chip implementation of the complete CDMA modem possible The design of a single chip CDMA Modem essentially calls for a high degree of cooperation between the Digital Signal Processing specialists for the architectural and algorithmic design and of VLSI design experts for the design of the chip layout

The CDMA system essentially consists of a base station system and a mobile station system Full duplex communication is possible via a forward link (base station to mobile) and a reverse link (mobile to base station) Fig 2 1 shows block diagram of a CDMA Modem showing the various stages involved in the forward link (demodulation) and reverse link (modulation)

For the forward link in the mobile digital processing occurs when data flows from the A/D converter to the demodulator where received data are demodulated and multipath combination is performed and finally to the deinterleaver to reestablish the original data ordering then to Viterbi decoder where the symbols are soft decoded and error corrected

For the reverse link in the mobile voice data are encoded in the vocoder sent to the modulator where the data are convolutionally encoded, and interleaved to counteract correlated deep fades scrambled direct sequence spread, and FIR filtered These data are then finally sent to a D/A converter and are subsequently RF modulated and transmitted

2 2 CDMA Modem Architecture

2 2 1 Forward-Link demodulation

The forward link demodulation system shown in Fig 2 1 has been redrawn in Fig 2 2 The figure shows various stages, a searcher, demodulating fingers, a symbol combiner, a frequency error combiner, power control, a deinterleaver, and a Viterbi

decoder included in the forward demodulation path in a CDMA Modem

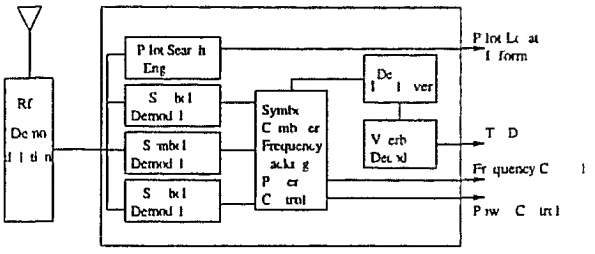


Figure 2 2 Forward link demodulation

The searcher allows the Modem to detect and lock onto a pilot signal on the forward link and to continually search for other pilots (from other base stations) that may have better signal levels

Demodulation is performed using a three finger RAKE type receiver Each of the fingers is capable of demodulating a component of a multiple path signal tracking a single pilot by itself and maintaining its own timing reference independent of other fingers At the output of each finger is a time deskew buffer that allows it to time track an average of four symbols away from other fingers The symbols from each finger are realigned and then combined in the symbol combiner to form soft decision data

The frequency error combiner combines the error signals from each finger, and uses the measure to bias the frequency of the local oscillator

The mobile station modem uses two methods to control transmit power *open loop power control* and *closed loop power control* With the first method the Modem attempts to estimate the path loss on the forward link and based on these measurements estimates the power required to transmit on the reverse link With the closed loop method reverse link power control information is sent in-line, at specified intervals, with the normal symbols being transmitted on the forward link Though this method is more accurate, it is not suitable for rapid adjusting of the mobile transmit power

The resulting symbol stream is passed on to a block deinterleaver where

the soft decisions are deinterleaved The deinterleaver spans either 20 or 80 ms depending on the rate of information being transmitted on the forward link

The deinterleaved symbols are passed on to a Viterbi decoder for an optimal decoding of the symbol stream The Viterbi decoding because of its wide applications in the communication fields and high hardware complexity will be our focus in this thesis

2 2 2 Reverse-Link modulation

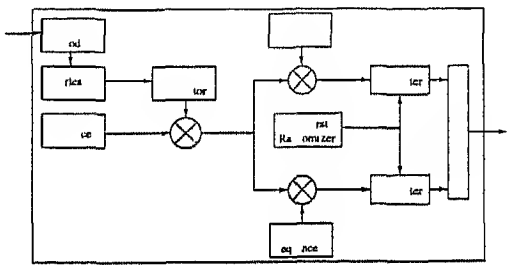


Figure 2 3 Reverse link modulation

The time tracking loop in the symbol combiner is responsible for maintaining reverse link timing which is synchronized with forward link timing The symbol combiner triggers the modulation of each frame of data to be transmitted on the reverse link Prior to each frame boundary a packet of data is written into the Modem Modulation of these data includes encoding interleaving scrambling, spreading, and filtering (refer Fig 2 3) For each 20 ms frame a data packet is transmitted by bursts into the input buffer of a convolution encoder of rate 1/3 and constraint length K=9

The code symbols are passed through a block interleaver which spans a 20-ms frame The interleaving algorithm forms an array with 32 rows and 18 columns At 9600 bps, the interleaver forms a 32×18 matrix as in Table 2 1

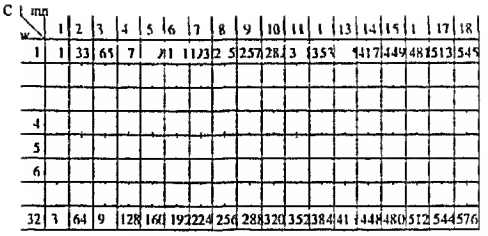


Table 2 1 Interleavıng algorıthm

At 9600 bps the transmıssıon sequence ıs to send ıow bv row ın a sequence order upto ıow 32 At 4800 bps the transmıssıon sequence ıs to send by the unıque order of rows as follows

$$J \quad J+2 \quad J+1 \quad J+3$$

for $J = 1+4ı$ and $ı = 0\ 1\ 2\ 3$,(32/4 1)

At 2400 bps the transmıssıon sequence ıs by a unıque order of rows as follows

$$J \quad J+4,\ J+1,\ J+5 \quad J+2 \quad J+6 \quad J+3,\ J+7$$

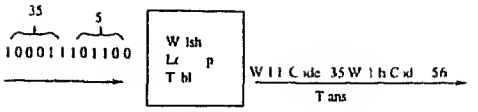
for $J = 1+8ı$ and $ı = 0\ 1,2,$,(32/8 1)

At 1200 bps

$$J+8,\ J+1,\ J+9,\ J+2,\ J+10,\ J+3 \quad J+11,\ J+4,$$
$$J+12,\ J+5,\ J+13 \quad J+6,\ J+14 \quad J+7,\ J+15$$

for $J= 1+16ı$ and $ı = 1,2$

The ınterleaved symbol stream ıs then converted ınto a set of orthogonal functıons Each streaın of sıx ınterleaved code symbols ıs mapped to a 64 bıt Walsh functıon



This stream of Walsh chips is then scrambled by BPSK spreading it with a pseudorandom (PN) sequence of bits produced by a 42 bit pseudonoise sequence generator

After the modulated symbol stream is BPSK spread it splits to form an I phase and a Q phase path for OQPSK signaling The I and Q phase paths are each spread with an independent PN sequence produced by two 15 bit PN generators programmed with different polynomials

Reverse link transmission power is reduced by not transmitting the redundant symbols that are present for data rates slower than 9600 bps Symbols are read from the block interleaver in a fashion which collects redundant data into groups and a data burst randomizing algorithm uses the 42 bit PN generator to pseudorandomly gate off the redundant data stream during transmission

Two 48-tap finite impulse response (FIR) filters are included in the modem one for the I phase data stream and one for the Q phase data stream These filters attain a stopband rejection of 42 dB The outputs of the two filters are truncated to 8 bits before multiplexing to a common output pin

This completes our discussion of a CDMA Modem In next chapter we will introduce techniques involved in implementation of a Viterbi decoder Viterbi decoding and the architecture of a Viterbi decoder will be our focus in the remaining chapters

Chapter 3

Implementation of Area Efficient Viterbi Decoders

In this chapter we describe techniques for implementing a VLSI Viterbi decoder The design implementation can vary between two extreme forms a completely parallel approach based on a single ACS unit for each node and a totally serial approach based on a single ACS unit for all the nodes We will examine the method employed for in place computation of the path metrics Also, the two forms of memory organization – register exchange and traceback method will be compared Our main objective will be to optimize our design for minimum area without sacrificing the additional features of a Viterbi decoder e g bit synchronization and bit error rate (BER) measurement

This chapter is devoted to the study of basic techniques employed in design of a Viterbi decoder, in chapter 4 we will review some of the state of art Viterbi decoder implementations and in chapter 5 we will describe our implementation of the Viterbi decoder based on the discussions in chapter 4 and 5 and that best suits our specifications

3 1 Introduction

The Viterbi Algorithm employed for decoding convolutional codes was discovered in 1967 Realized as an application of dynamic programming Viterbi algorithm has been used for estimation and detection problems in digital communications Recently it has been used in speech recognition where characters are modeled as hidden Markov models Earlier implementations of the Viterbi decoding algorithm were based on discrete circuits The size and complexity of these circuits limited the application of Viterbi algorithm to only deep space applications and to very expensive large digital communication systems [15]

The Viterbi Algorithm may be viewed as a solution to the problem of maximum *a posteriori* probability estimation of the state sequence of a finite state discrete time Markov process observed in memoryless noise A tutorial on Viterbi algorithm can be found in [16]

Recently a number of VLSI architectures to implement the algorithm have been suggested Broadly these architectures can be divided into two categories

- node parallel and
- node serial

The Viterbi algorithm requires computation of a path metric for all encoder states (2^{K-1}, K=constraint length) for each timing instant

The constraint length K and the coding rate R of the coder–decoder are two important parameters that determine the coding gain Since the number of states in the trellis equals 2^{K-1},the complexity of the Viterbi decoder doubles for every single increase in K Although a higher value of K yields a larger coding gain (4 8 dB for K=9) the achievable gain is limited by the speed and complexity of the decoder

Assuming equiprobable input data sequences, the decoder chooses the path (i e the sequence x) that maximizes the log likelihood function, log P(y|x) The maximum likelihood function is difficult to implement efficiently, so typically the maximum likelihood decoder is implemented using distance measures In a binary symmetric channel the metric reduces to Hamming distance, while for eight level

soft decision it reduces to Euclidean distance Not more than eight levels of soft decision are used as eight levels of quantization results in a loss of less than 0 25 dB compared to infinitely fine quantization and therefore quantization to more than eight levels yields little performance improvement [15]

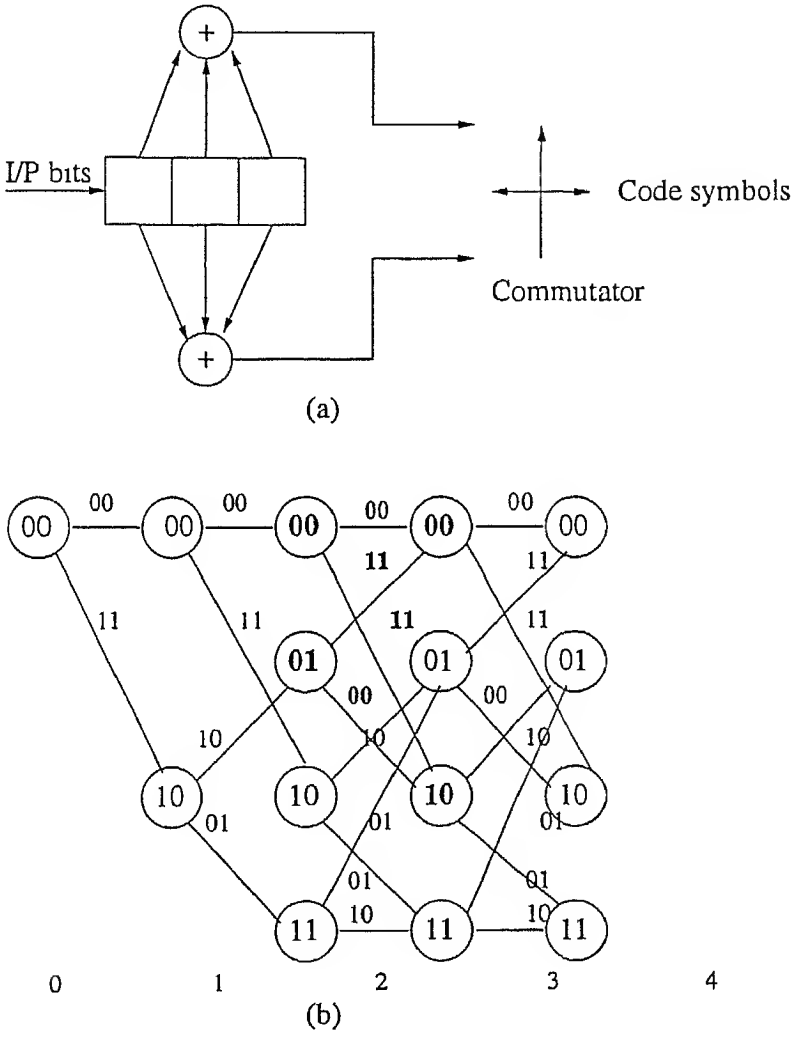


Figure 3 1 (a) K=3,R=1/2 Convolutional encoder (b) Code trellis diagram

The Viterbi algorithm, for memoryless noise, is a Markov process, and the probability $P(x_{k+1}|x_0, x_1, \quad, x_k)$ of being in state x_{k+1} at time k+1, depends only on the state x_k at time k

$$P(x_{k+1}|x_0, x_1, \quad, x_k) = P(x_{k+1}|x_k)$$

A transition from state x_k to x_{k+1} is denoted as ξ a state sequence $(x_0 \quad x_k)$ by x and a sequence of observations by z

The Viterbi algorithm can be defined as finding the state sequence x for which the *a posteriori* P(x|z) is maximum

In Fig 3 1 a description of Viterbi decoding known as *trellis* has been shown In a trellis each node corresponds to a distinct state x_k at a given time (k) and each branch represents a transition ξ_k to a new state x_{k+1} at the next instant time (k+1) For every possible state sequence there corresponds a unique path through the trellis Given a sequence of observations z we may assign a metric with x where x is the path in the trellis associated with z The metric chosen may be proportional to ln P(x,z) P(x z) maximum corresponds to P(x|z) maximum and there is a one to one correspondence between ln P(x z) and P(x|z) Hence we may assign a metric

$$\lambda(\xi_k) \equiv -\ln P(x_{k+1}|x_k) - \ln P(z_k|\xi_k)$$

for each transition

In the trellis a state sequence $(x_0, x_1 \quad x_k)$ corresponds to a path starting at the node x_0 and terminating at x_k There will be in general, several such path segments, each with some metric

$$\lambda(x_0^k) = \Sigma_{i=0}^{k-1} \lambda(\xi_i)$$

The shortest such path segment is called the survivor corresponding to the node x_k and is denoted as $s(x_k)$ The shortest complete path s begins with one of these survivors Viterbi algorithm finds the shortest route in the trellis At any time we need remember only 2^{K-1} survivors and their lengths $\Gamma(x_k) \cong \lambda[x_k]$ To get to time k+1, we need only extend the survivors at time k by one time unit, compute the metrics of the extended path segments, and for each node x_{k+1} select the shortest extended path segment terminating in x_{k+1} as the corresponding survivor at time k+1

Certain modifications become necessary, first, the input sequences received are very long, or infinite, and the survivor length (decoder memory) needs to be

truncated to some manageable length d Also if k (time instant) becomes large the metrics may grow indefinitely and it becomes necessary to renormalize $\Gamma(m)$

A closer look at the trellis reveals that the transitions in any unit of time can be segregated into $2^{\nu-1}$ disjoint groups of four each originating in a common pair of states and terminating in another common pair A typical such cell has been shown in Fig 3 2 The states at time k have been labeled as x'0 and x 1 and the states at time k+1 as 0x and 1x' (x' concatenated with 0 or 1 denotes the register contents at state k or k+1)

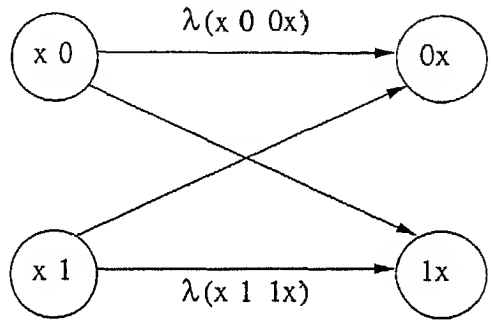


Figure 3 2 Cell of a shift register process trellis

A butterfly extracted from Fig 3 1 trellis diagram has been drawn in Fig 3 3

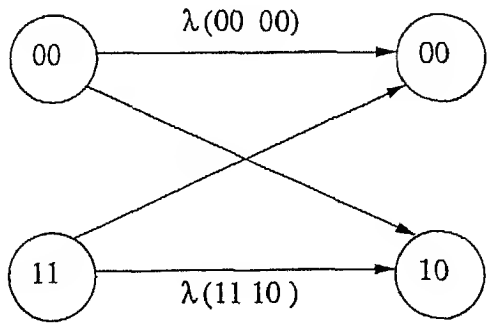


Figure 3 3 A butterfly extracted from Fig 3 1(b)

3 2 Implementation of a Viterbi Decoder

3 2 1 Memory organization for path metrics

Similar to in place computation of discrete Fourier transform we can have in place computation of path metrics in a Viterbi decoder which obviates the need for a double buffer

However a price that is paid for this saving of memory is that we do not have a straightforward addressing mechanism Let us consider that binary data is fed to a decoder with number of states equal to eight The decoder has eight hypotheses ending in 000=0 001=1 010=2 011=3 100=4 101=5 110=6 111=7 Two successors (refer Fig 3 2) of 000 and 001 are 000 and 100 This means that we read metrics from 0 and 1 and write them in 0 and 4 This is not in-place computation

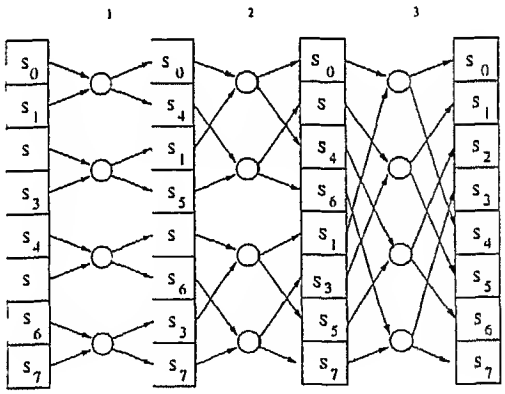


Figure 3 4 Evolution of contents of Path memory for in-place organization (K=3) for three decoding cycles

To put the metrics in place we devise an addressing scheme that changes after every decoding cycle In general, at a time instant i the metrics are found by generating their natural addresses but rotating the bits of these addresses by i (refer Fig 3 4) places before reading or writing the metrics from or into the memory A cyclic shift of i places is identical to a cycle shift of i modulo K (constraint length) places

A possible scheme for address generation has been shown in Fig 3 5

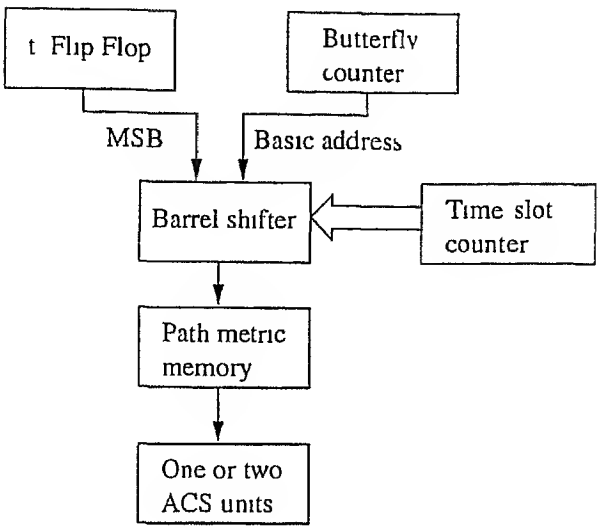


Figure 3 5 Address generation for in place computation

3 2 2 Accelerating metric computation

To employ some parallelism for the metric computation based on binary data one can divide the path metric memory in two blocks an even parity block and an odd parity block [18, 19] Thus, one metric will always be read from an even parity block and another metric from the odd parity block The even parity and odd parity blocks may be further be subdivided to further accelerate metric computation In general for M ary case each of the M possible modulo M digits will occur for one of the M addresses used together

3 2 3 Memory organization for survivor paths

Path memory processing consists of storing in path memories the d most recent decisions of the state processors and providing a mechanism for deriving the most probable transmitted information sequence from these decisions Two methods are commonly used for determining the transmitted sequence the traceback method and the register exchange method Most high speed very large scale application

specific integrated circuit (VLSI ASIC) implementations of Viterbi decoder employ one of these

Methods of reading the path memories

- Traceback

This requires that the 2^{K-1} decisions calculated for each symbol are saved in a path memory represented as a two dimensional array with symbol period identifying a column of decisions and state identifying a row of sequential decisions The traceback is started from an arbitrary state The path is then extended back through the trellis by recursively determining the sequence of the states that form the path It is necessary to traceback decision depth d to ensure that all paths have merged with the correct one A path memory of depth 4 or 5 times the constraint length is sufficient for negligible degradation from optimum decoder [15]

Traceback involves reading of columns of 2^{K-1} decisions and selecting one bit from each column indexed on the current state to update the current state and continue the traceback Higher speeds are met by implementing $2d$ tracebacks generating d valid decoded bits, while a further d bits are received at the decoder and processed by the ACS block

- Register Exchange

The register exchange method is based on the movement of hypothesised information sequences within the path memory This is facilitated by implementation of the path memory as a set of trellis connected shift registers Rather than storing in memory the column of decisions calculated for each symbol and then determining the path by subsequent access to the memory like in the traceback method, the register exchange method moves the hypothesised informati on sequences or paths from state to state as each received symbol is processed The new decision calculated by the ACS processor for each state is added to the beginning of the path currently associated with each state The merging of two paths at a particular state manifests itself as a duplication of the path up to that point in two different registers Just as

with the traceback technique a path of length equal to the decision depth must be determined before valid decoded bits are obtained

Traceback versus register exchange

With $2d$ tracebacks, generating d valid decoded bits traceback requires a total of $3d2^{J-1}$ bits of storage and has a latency of $2d$ Speed advantages higher than 2 1 can be used to reduce storage and latency A major cost of traceback technique is the considerable additional circuitry needed for RAM control decision bit selection and output buffering The RAM is operated as a circular buffer and the output bit buffer is essentialy a FIFO The additional circuitry consists largely of random logic

Implementation of register exchange requires the theoretical minimum latency, d symbol periods, and storage of $d2^{K-1}$ for a code with 2^{K-1} states The register size is equal to the decision depth d with one register being used for each state

The exact arrangement of the pipelined register exchange depends on the desired speed of operation In addition to trellis wiring and register cells, the register exchange process requires multiplexers to select surviving paths To achieve the highest possible speed in the path memory an instance of trellis wiring and multiplexers must be placed between each column of decision bits At reduced speeds several bits of register may be placed between each multiplexer and trellis connection

Register exchange offers the potential for very high decoded bit rates, while traceback is competitive for all other bitrates and is more closely linked to the current automated layout tools and memory technology that dominate the ASIC arena Implementation of the register exchange method becomes viable for longer constraint length code only through the use of the state labelling

3 2 4 State metric representation and normalization

Normally 4 or 5 bits are used for state metric representation To prevent an unbounded growth of the state metric, it becomes necessary to renormalize the state

metric from time to time

In the Viterbi algorithm only the relative magnitude of the state metrics is important As a result an offset positive or negative, can be added to all the state metrics in any symbol One of the techniques has been to reduce all state metrics by a certain amount in case they all exceed a certain threshold value For example, the MSB of every metric register is monitored, and if they are all one normalization occurs by removing the MSB of all metric registers One of the novel techniques suggested in [17] has been to use two s complement as an alternative to the normalization method The registers are designed to store metrics of length greater than $2D_{max}$, where D_{max} is the maximum possible difference between state metrics By using two s complement arithmetic the overflows can be accomodated Though it is computationally faster and gives an increased throughput the approach has been avoided for reasons which will be made evident in the next section

3 2 5 Bit synchronization and bit error measurement (BER)

A Viterbi decoder may have a synchronizing unit [20] for determining the beginning of the received symbol stream and if required to resolve the symbol polarity [1] A 1/2 convolution encoder encodes each binary bit into two symbols, and the output stream can be grouped into packets of two symbols as shown

$$, s_{k-1}s_k, s_{k+1}s_{k+2}, s_{k+3}$$

or

$$s_{k-1}, s_k s_{k+1}, s_{k+2}s_{k+3},$$

The synchronizer resolves between the two data streams

A synchronizing unit relies on an input error indicator for synchronizing the input data stream The input sequence errors can be indicated by either monitoring the rate at which the path metrics are increasing or by using the merging properties of the trellis A decoder *out of sync* tends to have surviving paths which merge slowly than a decoder *in sync* Any such condition when detected is fed to the

[1] where non transparent odes are used

synchronizing unit for correction

One possible circuit for monitoring input sequence error rate has been shown in Fig 3 6 Here the bit synchronization is a two step process A counter initialized externally by the microprocessor monitors the normalization rate The counter is clocked every time the path metrics are normalized The overflow output of this counter is used to indicate *out of sync* condition The external count fed to this counter sets an *in sync/out of sync* threshold for the Viterbi decoder Another counter also initialized externally, is clocked by decoded data output clock The overflow output of this counter is used to indicate *in sync* condition

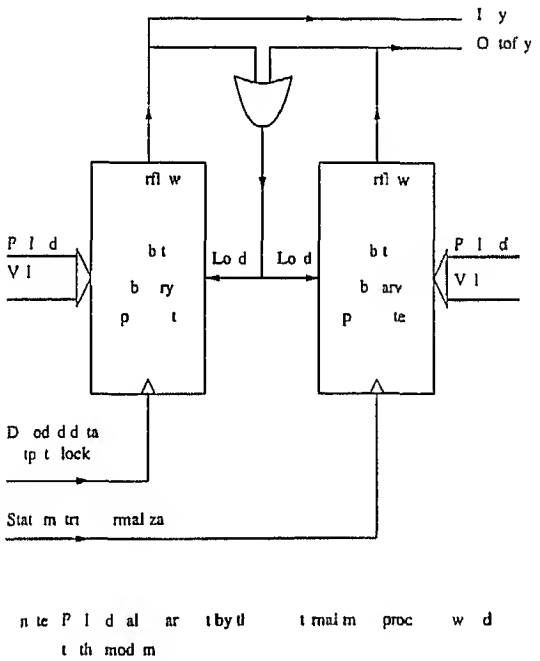


Figure 3 6 Normalization rate monitor circuit

The second step of the automatic synchronization process attempts to correct an indicated out of sync condition by offsetting the data input to the decoder prior to the actual decoding process

Twos' complement arithmetic destroys the reliability indicators, viz normalization rate or trellis merging information, that can be used in the synchronization loop for synchronizing the input sequence stream, and hence is not suited for Viterbi

decoder where bit synchronization unit forms an integral part of the design

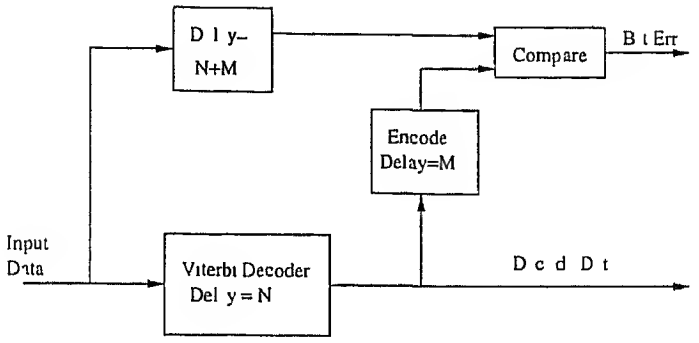


Figure 3 7 Re encode and compare circuit for channel BER estimation

By introducing a delay unit and a channel encoder which re encodes the decoded data stream from the decoder a comparison unit gives a measure of channel errors by making a bit by bit comparison of the re encoded data stream and the delaved input data stream A possible configuration for measuring channel bit error rate (BER) has been shown in Fig 3 7

This completes our discussion of the Viterbi decoding algorithm In chapter 4 we will compare the various techniques involved in the implementation of the Viterbi decoding algorithm

Chapter 4

Area Efficient Viterbi Decoder – Design Issues

In this chapter we will compare some of the approaches taken in the implementation of the Viterbi decoders However, our discussion will be confined to designs that have been fabricated and tested Wherever possible we will discuss the motivation behind a particular approach

4 1 A comparison of Different VLSI Viterbi Decoder Designs

4 1 1 Introduction

Broadly speaking the different realizations of Viterbi decoders may be divided into two categories, viz node parallel and node serial [21, 22 23] In the node parallel approach, computations for all nodes in the trellis are done simultaneously In the node serial approach, computations are done sequentially by the node and the de coding operation is therefore much slower There have been several implementations atempting at tradeoff between area of implementation and the number of nodes A number of ACS units, say P, where P is less than the number of nodes N, are shared

by the nodes in carrying out the branch metric computation One attractive feature of this method is that it allows a *linear* tradeoff between area and the speed [24]

For highest speed decoding a fully parallel processing implementation with one processor per decoder state is used These processors or ACS units decide which of the possible information sequences entering a state is more likely to survive The processors need to be interconnected according to the trellis diagram of the code this gives rise to a problem of different kind where reduction of the wiring area on a silicon chip is the main aim This will be discussed in the next section of the chapter The code memory depends on constraint length K as 2^{K-1} and this exponential dependence on code memory length forces a tradeoff betwen speed and constraint length of the convolutionally encoded code that can be implemented on a single chip To reduce global wiring which may occupy 37% area of the chip several designers have proposed locally connected processor architectures that completely eliminate global interconnects [29] and also reduce the number of ACS elements by processing more than one state per processor [30] For increasing decoder throughput multiple decoders have been run on either interleaved data or blocked data

Another other aspect that has been investigated has been the design of the path memory processing There are two methods in common use register exchange and the traceback method Register exchange method has been used in high speed ASIC design for 25 Mbit/s NASA (2,1,7) decoder and traceback has been used in 17 Mbit/s decoder developed by Qualcomm [21] There is still a disagreement as to which of the two methods gives the most efficient VLSI implementation

4 1 2 Algorithms and the architectures

VLSI grid model can been used to have an account of the chip area In the VLSI grid model a VLSI chip may be viewed as a computation graph whose vertices are called nodes and edges are called wires Nodes can be considered as processors responsible for the computation of Boolean functions The wires are simply electrical connections responsible for transfer of information between the nodes and also for distribution of power supply and timing information to the nodes The area of the

circuit is defined as the area in unit squares of the smallest rectangle enclosing the circuit Two schemes based on the shuffle exchange (SE) and cube connected cycles (CCC) are identified for the placement and routing of the required ACS processing elements Discussion and applications of shuffle exchange organizations can be found in [25] and [26]

The shuffle exchange graph consists of 2^{k-1} nodes each of which correspond to one state of the single stage trellis Each node labeled 0 to $k-1$ is associated with an equivalent $k-1$ binary string s_{k-2} s_1 s_0 Two nodes k and k' are linked via a shuffle edge or an exchange edge A shuffle edge links two nodes k and k' if k and k' differ only in the first bit i e $k = s_{k-2}$, s_1 s_0 and $k' = s_{k-2}$, , $s_1\bar{s_0}$ An exchange edge links two nodes k and k' if $k = s_{k-2}$ s_{k-3} s_1 $s_0$and $k' = s_{k-3}$ s_0 s_{k-2} The Shuffle Exchange graph can be related to a single stage trellis Assuming the number of states 2^{ν} the states may be labeled from 0 to $2^{\nu-1}$ respectively Each initial state identified by an equivalent ν bit binary label $x_{\nu-1}, x_{\nu-2}$, x_1, x_0 maps into two states $x_{\nu-2}$, , x_0 $x_{\nu-1}$ and $x_{\nu-2}$, , x_0 $\overline{x_{\nu-1}}$ The edge from an initial state to the two final states can be considered as a shuffle operation mapping state $x_{\nu-1}, x_{\nu-2}$, x_1 x_0 to $x_{\nu-2}$, x_0, x_1, or a shuffle operation followed by an exchange operation mapping the initial state to state $x_{\nu-2}$, , $x_0, \overline{x_{\nu-1}}$ Expressed in terms of *Shuffle* and *Exchange* the input node k is connected to nodes k' and k'' where

$$k' = Shuffle(k)$$

and

$$k'' = Exchange(Shuffle(k))$$

This interconnection structure can be implemented on a Shuffle Exchange network A cycle of shuffle edges in a Shuffle Exchange graph is known as a necklace The architecture is cleaved into cycles of shuffle edges to generate a layout for embedding the Viterbi algorithm in silicon

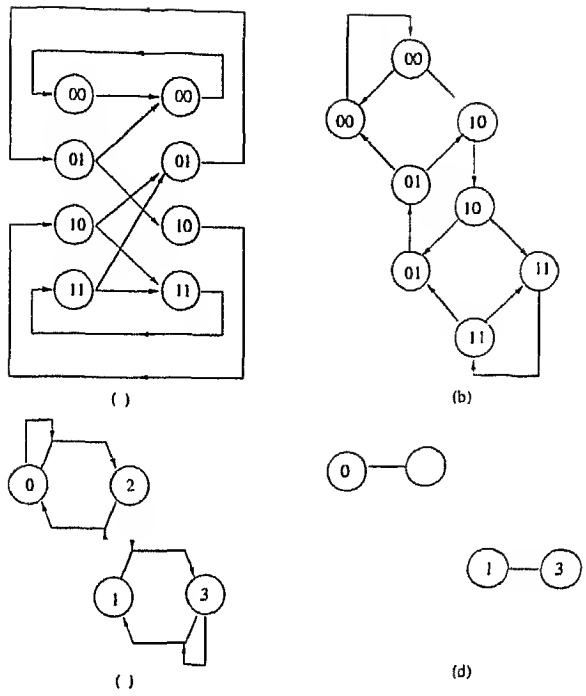


Figure 4 1 (a) Single stage trellis with recirculation edges (b) Graph unfolded (c) The Add Compare Select nodes in (a) separated (d) Grid model layout for the four node 2 SE graph

Fig 4 1 shows grid model layout for a single stage trellis with recirculation edges In Fig 4 1(b) the nodes have been arranged into a planar graph In Fig 4 1(c) only those nodes which serve as wiring distribution points have been split and stretched Fig 4 1(d) shows a 2 SE[1] that has been generated for embedding the Viterbi algorithm The necklace has been drawn as a dashed rectangle consisting of two long vertical segments and two unit length horizontal segments The two exchange edges have been shown as solid horizontal lines

The second architectural strategy proposed is based on the cube connected cycles (CCC) However, CCC results more interconnect wire area than the SE ar chitecture

It has been observed in [27] and in [28] that applying the optimal shuffle

[1]since the number of nodes is a power of two the graph is referred as 2 SE (Shuffle-Exchange) graph

exchange layout proposed in [29] yield layouts with poor utilization of silicon area as such layouts cannot be compacted This is as a result of constraints imposed by abutting nodes and the distribution of global signals (buses clock signals and power supply) which are irregular and hence take greater silicon area

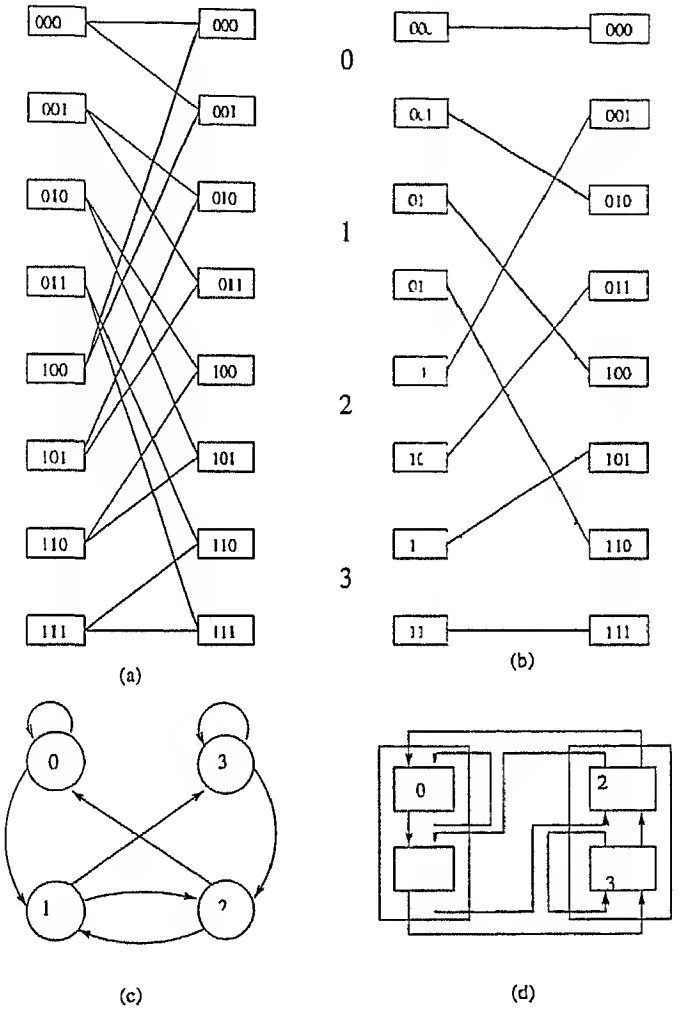


Figure 4 2 Steps leading to ring topolgy (a), (b) organization in pairs, (c) Identification of Hamilton cycles, here (0,1,3 2,0), and the (d) Resulting two column floor plan

In [28], it has been suggested that the ACS elements be arranged to form one directed cycle, in which one of the two outgoing path metric buses of an ACS elements connects directly to the neighbor The ring has been organized as two

columns as shown in Fig 4 2 Based on the concept of a ring organization a Hamilton cycle[2] is found for the *deBruijn* graph

The two ring topolgy has the advantage of reducing the routing of the buses into a channel routing problem Further the regularity of the structure allows easy vertical routing of the global signals Determining Hamilton paths is a NP complete problem The application of this concept is therefore limited to 32 ACS units Beyond that value the computational complexity prevents an exhaustive search of all the Hamilton cycles

The throughput of a Viterbi decoder can been increased by applying lookahead to both ACS and trace back recursions [31] A 2^{K-1} state trellis can be iterated from time index n k to n by decomposing the trellis into 2^{K-k-1} subtrellises each consisting of k iterations of a 2^k state trellis Each 2^k can be collapsed into an equivalent one stage radix 2^k trellis by applying k levels of lookahead to the recursive update

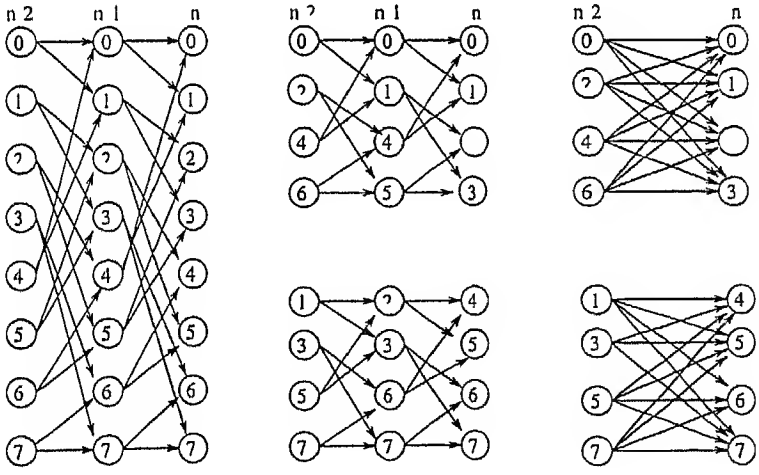


Figure 4 3 (a) 8 state radix 2 trellis (b) 4 state subtrellis decomposition (c) 8-state radix 4 trellis

An example of the decomposition for an eight state radix-2 trellis into an equivalent radix-4 trellis using one stage of lookahead is shown in Fig 4 3

[2]Hamilton cycle is a path which goes through all of the vertices once and only once [32]

If the delay through a 2^k and 2 way ACS are equal a possible k fold speedup is achievable for a complexity increase of 2^{k-1} in the ideal case However the actual speedup obtained has not been mentioned in the original paper [31] It has been observed that radix 4 case corresponds to the best tradeoff between area complexity and the speedup This result is technology dependent and technologies better than 1 2 μm may see higher radix lookahead implementations

In addition to the total serial and total parallel approach there have been attempts to have mixed designs A bit serial architecture with parallel nodes has been given in [33] The node parallel architecture carries computations in a bit-serial manner The reason for using bit serial manner has been to reduce wiring requirements and to allow interconnections of the chips for larger constraint lengths

4 1 3 Path memory processing

As mentioned the two common methods of path memory processing viz register exchange and traceback method have been used in the design of high speed decoders Latency constraints in traceback method have been reduced in hardware by dividing symbol time into multiple reads of the past decisions and one write of the current decisions In a hardware implementation generally, $2d$ tracebacks are performed, generating d valid decoded bits, while a further d bits are received at the decoder and processed by the ACS unit Register exchange because of its very nature, introduces the cost of trellis wiring However it is possible to reduce trellis wiring by relabelling the states within the region The total reduction of wiring area using this technique can be as high as 66% [27]

4 1 4 ACS arithmetic and normalization

The number of bits used to implement the ACS arithmetic depends on the maximum path metric range The intricate process of normalization can be avoided by using twos complement format and ignoring the overflow This approach has been used in [31], where the number of bits required to implement state metric has

been obtained as

$$\Gamma_{bits} = \log_2 b_{max} + k \lambda_{max} + 1$$

Where $b \leq \lambda_{max} log_2 N$ is the bound on the maximum dynamic range of state metrics N is the number of states and λ_{max} is the maximum branch metric for the radix-2 trellis As has been discussed in chapter 3 twos complement arithmetic leads to the loss of indicators used for bit synchronization and invariably all commercial decoders use the normalization rate to automatically synchronize the input data

In [34] the most negative state metrics are preserved however to prevent the state metrics from going too negative a positive offset is added to all the metrics to preserve their relative magnitude To prevent positive overflow large positive state metrics are hard limited To avoid information loss which can result if all state metrics are hard limited, a negative offset is added to all the state metrics if they all move away from the negative end

In [28] a dynamic solution to the state metrics normalizations is given Path metrics have been represented by 7 bits and branch metrics by 4 bits A global precharged wire exists that any ACS can discharge if its path metric is less than 32 The signal is input to a register in each ACS element, and this register drives the three most significant bits of the branch metric operand to the adder Rescaling is initiated if the minimum path metric exceeds 32

This concludes the discussion on the various techniques that have been used for fabricating Viterbi decoders as VLSI chips In the next chapter we discuss our implementation of a 19 2 kbps, 256 state Viterbi decoder We will also discuss the suitability of Field Programmable Gate Arrays (FPGAs) for obtaining a rapid prototype of a Viterbi decoder

Chapter 5

Design of 19.Kbps, 256 State, Area Efficient Viterbi Decoder Using High-Level Synthesis

5 1 Introduction

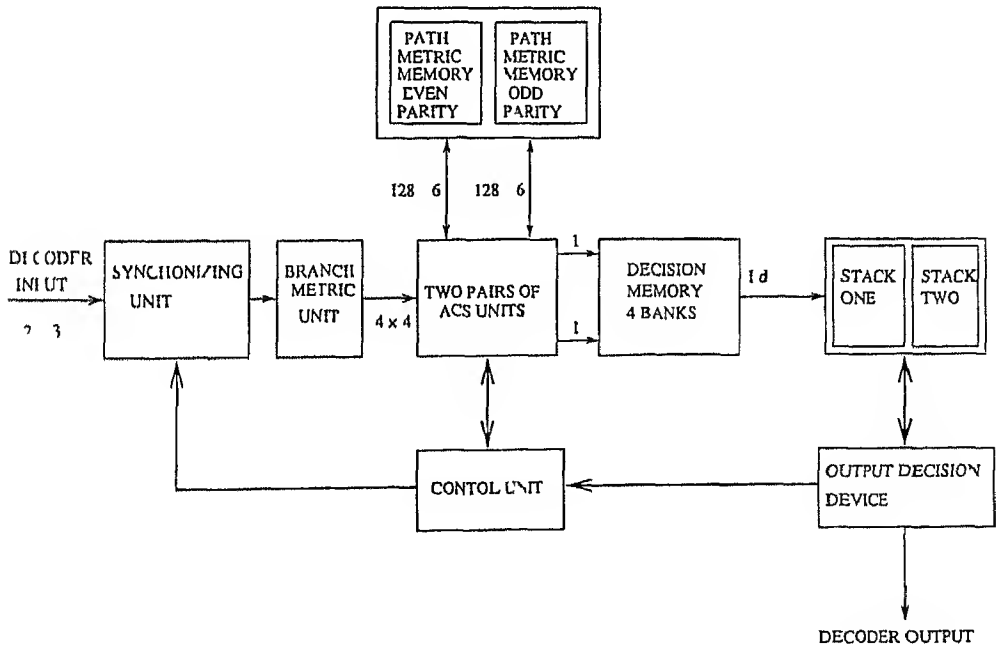


Figure 5 1 Viterbi decoder block diagram

A major contribution of this thesis has been in the design and implementation of a Viterbi decoder A Viterbi decoder is a very important block in any CDMA modem The aim was to design a 19 2 kbps 256 state Viterbi decoder with added capability of catering to higher input data rates In the literature, the discussions have been primarily focussed towards the analysing of a proposed architecture rather than towards its implementation, this is probably due to the proprietary nature of the final implementation To the best of our knowledge none of the existing literature discusses Viterbi decoding implementation based on high level synthesis targetted for Field programmable Gate Arrays (FPGAs) This has been the focus in the present thesis

Besides the above, the issues of the organization of a path memory, decision memory, the decision memory reading techniques and the clocking mechanism have been left unanswered Some of the implementations have used twos complement representation for the path metrics However, this is achieved at the cost of destroying an important information used for input bit sequence synchronization The aim was to retain this bit synchronization information, even though it made the normalization of the path metrics essential Normalization of the path metrics is a difficult operation and to implement this feature there are two options– either to use a special time slot for this operation or else to use very fast subtracters As reserving a time slot slows the decoding operation and hence the decoding speed the latter option was chosen

In this thesis we explore an implementation methodolgy for rapid prototyping We chose FPGAs because of their reprogrammability and short programmability times With FPGAs it is possible to immediately assess the impact of various design changes on the final architecture However, designing with coarse grained FPGAs such as Xilinx FPGAs, which we have used in our implementation is totally different from standard cell based or mask programmable based ASIC designs

In this thesis we highlight the possibility of efficiently implementing in FPGAs very complex digital systems with stringent design constraints, such as Viterbi decoder The implementations are efficient both in terms of the number of logic

blocks used and also in terms of the timing constraints which need to be satisfied

In designing the Viterbi decoder we have used some of the features discussed in the literature however we have attempted to compare their merits with other existing techniques and also the advantages that have accrued because of their use It has not been our attempt to give just another implementation of the Viterbi algorithm but to create design guidelines which may be used in any future implementation of the decoder The methodolgy that we have tried to develop doesnot limit its usefulness to only the Viterbi algorithm but can be used for the rapid-prototyping of any Digital Signal Processing algorithm

We have used Verilog a high level hardware description language for the RTL description of our Viterbi decoder We found that writing a synthesisable code was totally different from writing the simulation model needed for validating and verifying the intended architecture We found that synthesisng tools support a subset of the original language features This can necessitate a total rewriting of the earlier RTL description We have also highlighted a few techniques that we have developed to circumvent some of the limitations of the synthesis tool Pinnauq It has been our experience that synthesis tools which support RTL description perform better than synthesis tools which support behavioral descriptions

In this chapter we will discuss a design flow by which we can closely relate the explored architecture and the VLSI implementation Central to this methodology is the synthesis of the logic (the VLSI implementation) from the code written in Verilog (abstraction of the architecture) The synthesis is performed in an exploratory manner, in order to arrive at an architecture solution with improved performance in terms of area and delay constraints

One of the advantages of writing the RTL description in Verilog was the inbuilt random number generators which return integer values distributed according to the various probablistic distributions e g , *Poisson_distribution(), Uniform_distribution()* [35] These functions were used not only to generate the input data stream while testing the proposed architecture, but also to simulate the noise bursts that affect the real time data transmission These functions return the same value given

the same seed and this feature facilitates debugging of the design by making the system inputs repeatable In the decoder the 8 level quantized data is decoded optimally by the Viterbi algorithm Two ACS pairs have been used to process the data serially Our decision for two pairs was motivated by a desire to reduce the total amount of logic gates used However using a single ACS unit was found to increase the interconnections and also the benefit of the butterfly structure of the Viterbi algorithm was lost Decisions from the ACS units are stored in a partitioned internal path memory A trace back through the path memory of depth 64 outputs a single data bit The path memory has been partitioned into even parity and odd parity memories The main drawback of this configuration is the complicated memory decoding structure However, this is more than offset by the reduction of other interconnection resources and the increase in the resources utilization Fig 5 1 shows the top level block diagram of the Viterbi decoder The implementation of the various blocks has been discussed in the following sections

5 2 Design Methodology

The algorithms have been specified in Verilog, a hardware description language The Verilog code gives the register transfer level (RTL) description of the architecture Data generators and pseudo noise genertaors written in Verilog are used to specify the inputs during the simulation of the architecture The inbuilt functions of Verilog are used to check the coding gain and hence the suitability of the architecture High level description allows to verify the suitability of various approaches e g in computation of the branch metric, the address for starting the traceback, the size of the traceback memory, etc , Implementation into hard wired architecture is done in an exploratory fashion This is done by a juxtaposed simulation of the DSP system specification and the RTL representation

Fig 5 2 shows the proposed methodology used for the synthesis and analysis of Viterbi the decoder circuit To create an FPGA implementation the initial Verilog file was modified to be acceptable to our Verilog FPGA interfacing tool Pinnauq

The Verilog code is compiled together with the Xilinx technology libraries and the result is saved in the Xilinx netlist format (XNF) file format XACT the Xilinx developmental system is then run on the XNF file to create a configuration file for the FPGA The performance of the synthesised design is obtained and this information is then used to hand tune the Verilog RTL code of the Viterbi decoder so as effect the desired change through Pinnauq

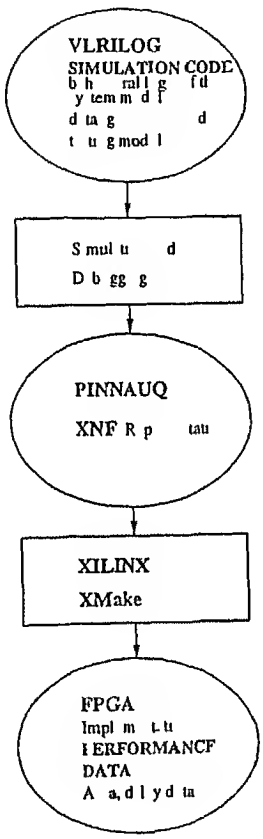


Figure 5 2 Proposed FPGA design flow

5 3 Decoder Implementation

5 3 1 Specifications

The specifications chosen for the Viterbi decoder to be implemented are as follows

1 K (constraint length) = 9 (256 states) R = 1/2 convolutional decoder

2 Generator polynomials $g^{(1)} = 561$ and $g^{(2)} = 743$ γ(coding gain)=4 77

3 8 level soft decisions inputs

4 Survivor path length 64

In addition to these specifications the decoder should have the capability of self synchronization in case the synchronization is lost A microprocessor initializes the decoder it also sets a threshold difference between the largest and the smallest path metric A self synchronizing circuitry takes a corrective decision if the difference between the two metrics does not exceed the user defined threshold Moreover the microprocessor can read the rate of the growth of the metrics before initiating any corrective measures

5 3 2 Branch metric unit

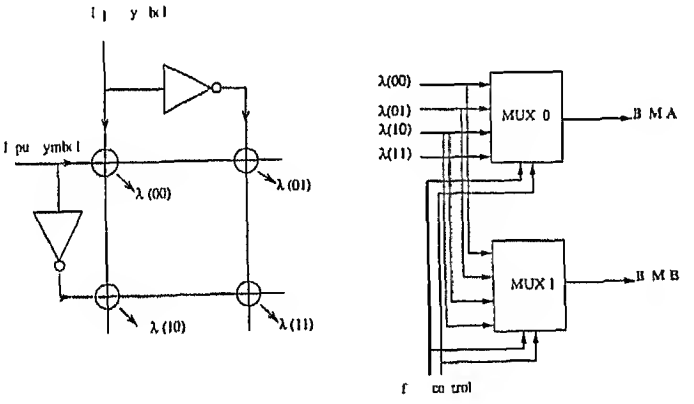


Figure 5 3 Block diagram of a branch metric unit

The two 3 bit soft decisions are combined to form four possible 4 bit branch metrics i e , $\lambda(00)$, $\lambda(01)$, $\lambda(10)$ and $\lambda(11)$ The branch metrics are generated as the Hamming distances of the input data stream from (00), (01), (10) and (11), for example Hamming distance of a bit stream '101000' from symbol '10' is 2 and from

the symbol '01'[1] is 12. The maximum value the Hamming distance can take is 14 and that explains using four bits for the branch metric. The block diagram of the branch metric unit has been shown in Fig 5.3.

While the path metrics are being generated, the branch metric unit is used to generate the branch metrics of the two new 3 bit data stream. The four bit path metrics are stored in four 7 bit registers (the three MSBs of the registers are made zero).

5.3.3 Add compare select unit

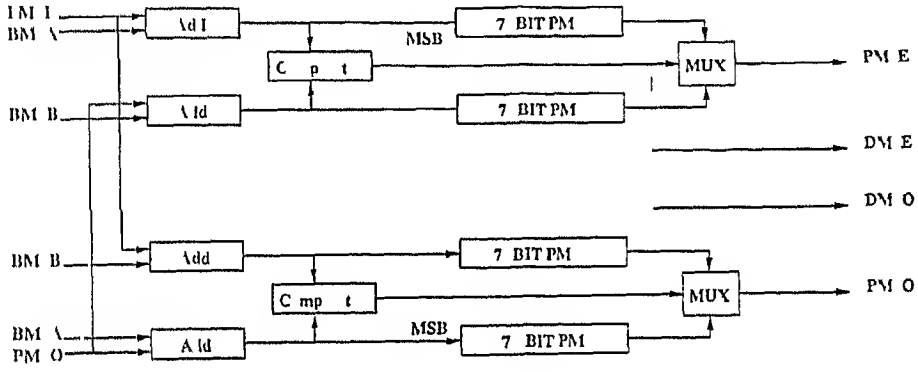


Figure 5.4 Block diagram of the ACS units

Area efficient 2 way ACS units have been designed using serial one bit adders (refer Fig. 5.4). The two comparators have been implemented by using serial one bit subtractors, it ideally suits our requirement as it results in a pipelined structure. The outputs of the subtractors are lateched into two 7 bit registers. The MSB of these registers are sent to the decision memory and are also used to choose between the two competing paths. The adder outputs are held in four 7 bit registers. The MSB of the surviving path metric is used to detect whether normalization is required or not. The path metrics are stored in the path metric memory with each location of length 7 bits. A control unit holds the value of the smallest and the largest path metric in two registers, which are accessible to an external microprocessor.

[1]Symbols '10' and '01' in 3 bit quantized levels are equal to 111000 and 000111 respectively.

The normalization is initiated only during the next iteration Normalization is done by subtracting the smallest path metric from the path metrics routed to the ACS units during the next iteration

5 3 4 Path metric memory

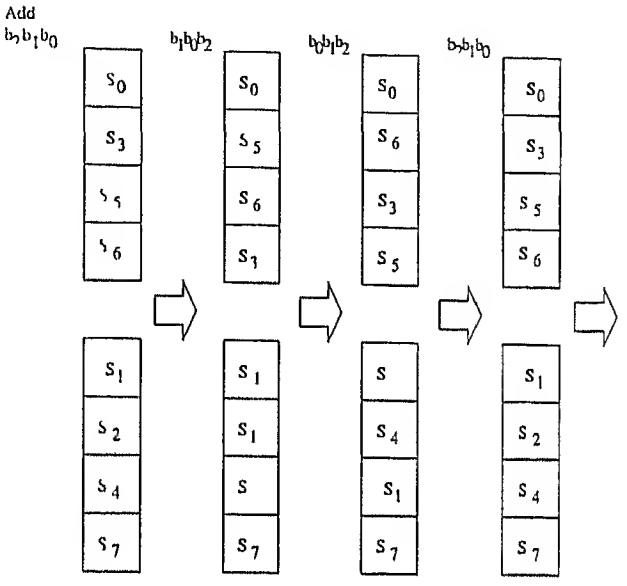


Figure 5 5 Address generation for in place computation of the path metric memory contents

To access benefits from the butterfly structure of the Viterbi algorithm, the path metric memory has been divided into two blocks viz even parity block and the odd parity block of sizes 128 × 7 each This allows parallel access of the two path metrics required for the computation of the next path metric The control unit scrambles the path metric memory addresses before accessing the memory, this allows inplace computation eliminating the need for a temporary buffer Fig 5 5 shows an example illustrating our concept

5 3 5 Decision memory

The traceback memory has been organized as a two dimensional structure, both the rows and columns can be accessed Fig 5 6 shows the organization of the decision

memory The number of the rows is equal to 2^7 Each column corresponds to one stage in the trellis

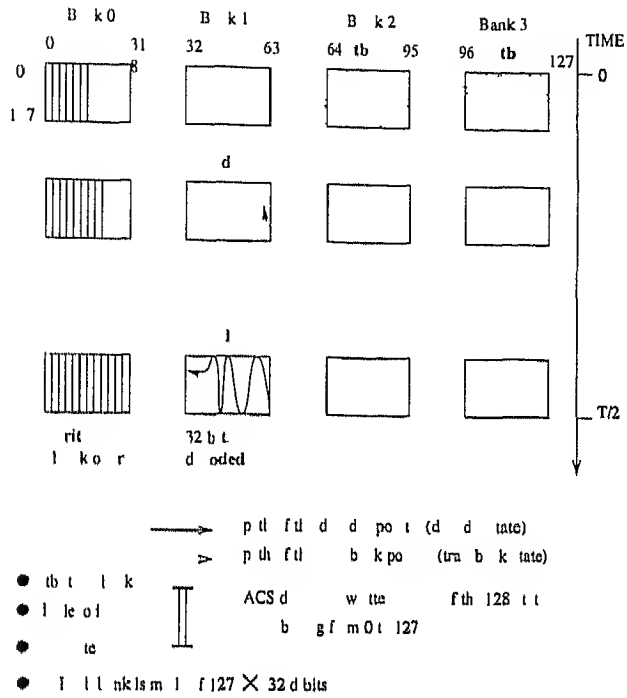


Figure 5 6 structure of the decision memory

The total memory has been divided into 4 banks with each bank of size $2^7 \times 32$ dibits[2] (the decision memory may be viewed as a three dimensional organization) Power consumption is reduced at the architectural level by beginning each cycle with the block address decoding and then enabling the two blocks (one for reading and the second for writing) This subarray architecture not only, results in a faster implementation, but also consumes less power, since the number of banks active at a time is limited to two and hence less capacitance is switched per read/write cycle The only drawback is the area penalty due to the increased overhead of the decode logic, the control logic and their routing interconnections [36]

The data is decoded in bursts The data is available after a traceback through two banks, and during the traceback through the third bank, the data i

[2]Each dibit is a group of two bits

decoded at the rate of 3 bits per input symbol The decoded data is written into a two stack structure which performs both the bit reversal as well as the elimination of the bursts The first decoded symbol appears after the first 128 bits have been processed by the decoder thus, making the output latency equal to 128 A single stack structure can be used if read and write operations are interleaved or a dual port memory is employed for the stack However it complicates the circuit design and hence was replaced by a double stack structure

5 4 Clock Mechanism

A single phase clocking scheme was preferred because it results in a simpler design style with reduced routing overheads Clock signals which drive each module are obtained from a common source and routed inside the chip to these modules The common source clock operates at a high frequency and the slower internal module clocks are obtained from counters which divide the common source clock

FPGAs, because of their fixed geometric structures, lack the flexibility of other ASIC implementation technologies with respect to the clock distribution scheme To reduce *clock skew* it is possible to distribute the clock pins during placement of the logic blocks such that the load capacitances in the clock distribution tree are balanced This is possible with Pinnauq by defining a constraint file while creating the netlist file (XNF file)

5 5 Rapid Prototyping of the Viterbi Decoder with XILINX FPGA

The Viterbi decoder was implemented using the Xilinx XC4000 [37] (Appendix B) FPGA series FPGAs allow shorter design turnarounds and reduced verification

times, which can potentially result in large design savings XC4000 series was chosen not only because it is supported by our synthesis tool Pinnauq a high level synthesis tool but also because of its advanced and flexible architecture The generation of the XNF description (or netlist) of our design was done with Pinnauq

Once the proper XNF description has been generated this mapping to the XC4000 series is straightforward using the Xilinx supplied tools for file format conversion The design is translated to a logic cell array (LCA) [3] file from where a serial bitstream file for configuring the CLBs is generated

The Xilinx XC4000 series contains special purpose hardware to efficiently implement fast carry logic as found in adders subtracters, counters and other re lated function blocks Normally little algorithmic advantages can be gained by substituting a superior description for such a function block It is also difficult for HDL compilers to use special purpose features which are available in FPGAs under certain conditions such as fast carry look ahead logic or the builtin RAM

Choosing the right implementation style for FPGA design can cause a huge difference in resource utilization We tried various algorithmic descriptions for our functional blocks e g serial one bit adders and subtracters The results obtained are given in Table 5 1

Description	FMAPS	HMAPS	IOBS	Other Resources
ADD (XOR)	2	0	6	0
ADD (+)	1	0	6	0

Table 5 1 Resource utilization for different styles

Xilinx provides a partial solution to this problem by supplying a library of adders, subtracters, counters, and comparators In Pinnauq, these libraries are used to implement complex functional units

[3] XC4000 in our case

Pinnauq uses X BLOX library functional units [38] by including references to the X BLOX modules in the Verilog code During the processing by XACT X BLOX is invoked as module generator to synthesize appropriate functional units

Fig 5 7 shows how a X BLOX description has been included in the Verilog description of one bit adders

```
module one_bit_adder(a b reset clock sum)
 input [0 0] a b reset clock
 output[0 0] sum
 reg [0 0] carry_in
 wire [0 0] sum
 wire [0 0] carry_out


 ADD_SUB_1_UBIN adder( A(a) B(b)
        C_IN(carry_in) FUNC(sum) ADD_SUB(1'b1) C_OUT(carry_out))

 always @(posedge reset or posedge clock)
  if(reset==1'b0)
     carry_in = 1'b0
  else if(clock == 1'b1)
     carry_in = carry_out
endmodule
```

Figure 5 7 Including X BLOX instantiation in a Verilog file

While using the X BLOX library helps optimize some circuits however most of the circuits cannot be optimized to use these features A solution is to include X BLOX Xilinx design library elements as *components* in Verilog and use either available circuits in the X BLOX libraries, or to generate one's own circuits with XACT or a schematic entry tool to be included in the design This feature was readily supported by Pinnauq and we optimized our circuits by including X BLOX design libraries in our Verilog description

Chapter 6

Conclusions And Future Work

6 1 Conclusion

Viterbi decoder is one of the most important blocks in a CDMA modem In this thesis we have designed and implemented the Viterbi decoder This has been targetted for FPGA implementation The methodology that has been developed is not specific to just the Viterbi decoder but can be used for implementation of any other Digital Signal Processing (DSP) algorithm We believe the same approach can be used in the design and implementation of the entire CDMA modem chip Some methods for accelerating the decoding rate have been developed and these methods can be used as guidelines in any future implementation of Viterbi decoder Our aim was to develop an area efficient 19 2 kbps, 256 states Viterbi decoder, but the design developed can cater to higher input data rates Some of the existing features such as input bit sequence synchronization, bit error measurement and self testing features have been incorporated

6 2 Future work

The implementation methodology used in the design of a Viterbi decoder can be extended to the complete design of a CDMA modem chip The design was targetted for FPGAs FPGA area optimization techniques are different from ASIC based

design optimizations techniques This feature of the FPGA design needs to be closely looked into Presently there are no guidelines available for writing HDL descriptions which can result in an optimal FPGA implementation We believe that this will be very much dependent on the synthesis tool used and on the FPGA architecture For example, using Xilinx X BLOX libraries seems to result in an area efficient implementation however the impact of including X BLOX descriptions in the high level RTL description needs to be closely assessed Minor restructuring of the RTL description can lead into tremendous variations in terms of the number of logic blocks used This feature needs to be investigated

The Viterbi decoder design because of its memory requirements needs more than one FPGA for its implementation and this necessitates partitioning of the design Because of high interconnection delays which can slow the decoding rate partitioning a single design into a multi FPGA targetted design is a real challenge

Appendix A

Spread Spectrum Techniques.

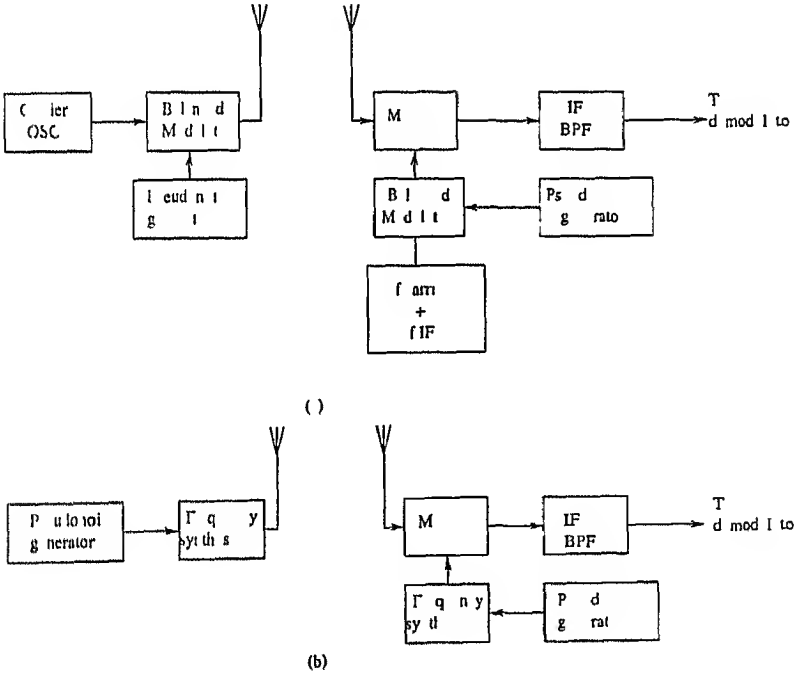


Figure A 1 (a) Direct sequence system (b) Frequency hopping system

A spread spectrum system is one in which the transmitted signal is spread over a wide frequency band, much wider, than the minimum bandwidth required to transmit the information being sent

Three general techniques are recognized as spread spectrum signalling methods

1 Modulation of a carrier by a digital code sequence whose chip rate is much higher than the information signal bandwidth Such systems are called direct sequence modulated systems

2 Carrier frequency is shifted in discrete increments using a pattern dictated by the code sequence These are called frequency hopping systems The transmitter jumps from frequency to frequency within some predetermined set, the order of frequency usage is determined by a code sequence

3 Pulsed FM or chirp modulation in which a carrier is swept over a wide band during a given pulsed interval

Appendix B

Using FPGAs for Application-Specific Digital Signal Processing Application - Viterbi Decoder A Case Study

B 1 Introduction

FPGAs are an alternative and a viable solution for implementing DSP algorithms which have traditionally been implemented using programmable DSP chips or hard wired application specific integrated circuits based on the standard cell or mask programmable gate array design methodology FPGAs combine the flexibility of a general purpose DSP and the speed, density, and low cost of an ASIC implementation The FPGAs, while retaining the advantages of a customized functionality that an ASIC provides, avoids the expensive developmental costs and the inability to make subsequent design modifications associated with ASICs

B 2 FPGAs Vs Traditional Approaches

Traditionally, DSP functions have been implemented using a general purpose Digital Signal Processor (DSP) or by using ASIC technology Whenever the application requires timing performances beyond the abilities of current generation Digital Signal Processors or when the expected volume justify a semi custom based solution these functions are implemented using standard cell based design methodology in a mask programmable gate array technology

However FPGAs provide an alternative solution that combines the best of both DSP and ASIC technologies without their respective limitations Like a general purpose DSP, FPGAs are programmable and can be modified *in situ* after production FPGAs have a flexible architecture that can be configured for a specific DSP function

B 3 Using FPGAs

FPGAs are pre wired circuits that are programmed by the users *in situ*, instead of during the fabrication This is achieved by downloading a user defined configuration in the form of a binary bit stream

The big difference between a gate array and a FPGA based ASIC implementation is that the user specific customizing fabrication steps are eliminated in FPGA A gate array undergoes atleast two steps in a silicon foundry This necessitates a careful verification of implementation design to avoid expensive design and foundry iterations

For FPGAs, a designer may specify his implementation by a netlist represented by a schematic diagram and then rely on an automatic place and route tool to map and interconnect components in the netlist implementation into the FPGA sites An alternative to using a schematic diagram for the design description is to use a Hardware Description Language (HDL) such as Verilog or VHDL Synthesis tools are then used to translate this HDL description into the netlist representation using components from the Gate Array's cell library

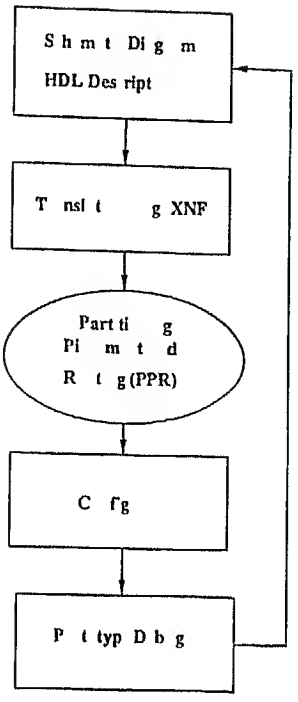


Figure B 1 FPGA design flow

Fig B 1 shows a typical FPGA design flow where the designer has specified the design with either a schematic or HDL description

B 4 Advantages of FPGAs

The advantages of FPGAs are

- Design upgradation does not require replacement of the FPGA Reprogrammability is the solution, some of FPGAs are reprogrammable
- Reprogrammable FPGAs can be dynamically reconfigured within the system Designs which can adapt to the changing conditions can be built with FPGAs Reconfigurability is a strong reason in favour of FPGAs
- In addition FPGAs avoid the need to develop accompanying test programs to catch manufacturing defects

B 5 Xilinx FPGAs

Xilinx FPGA devices possess the following features which enable implementation of high performance design [39]

- Flexible logic blocks with bit level arithmetic features - allows Distributed Arithmetic implementations of DSP algorithms
- Distributed RAM and ROM Increases operand bandwidth
- A register rich architecture enables a high degree of pipelining leading to increased performance

Configuring a Xilinx FPGA starts with a design, usually a block diagram or a schematic entered through a schematic capture tool The schematic is then automatically converted into the Xilinx Netlist Format (XNF) The XACT software in a Xilinx developmental system, first partitions the design into logic blocks, then finds a near optimal placement for each block and finally selects the interconnect routing The user has the flexibility to impose specific constraints on the above processes of Partitioning, Placement and Routing (PPR)

Once the design is complete it is documented in a Logic Cell Array (LCA) file, from which the FPGA configuring serial bitstream file can be generated Inside the device these configuration bits, control or define the combinatorial circuitry, flip flops, interconnect structure, and the I/O buffers

Xilinx also provides a module generator called X BLOX for its new gener ations of FPGAs (e g XC4000, 4000E etc ,) The module generator can be used to implement data path operators efficiently Besides X BLOX, Xilinx provides a library of macros known as Unified Library components to speed up the implemen tation process

Fig B 2 shows Configurable Logic Block (CLB) element of the XC4000 series Each CLB packs a pair of flip flops and two independent 4-input function generators A powerful and flexible CLB surrounded by a versatile set of routing resources contributes to a high cell usage in a Xilinx FPGA design

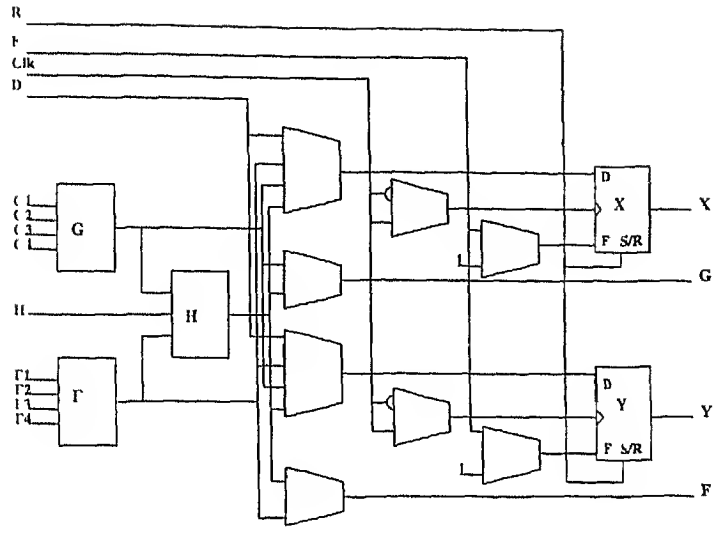


Figure B 2 The configuration logic block for the XC4000

B 6 Pinnauq - A tool for FPGA Design

Pinnauq is a high level synthesis tool for the Verilog HDL developed by Silicon Automation Systems (India) Pvt Ltd Bangalore The tool is meant for synthesising designs described in Verilog, and which are targetted for FPGAs The tools supports a subset of instructions available in Verilog

The version of the tool used during this thesis supports transformations targeted towards Xilinx FPGAs This version of Pinnauq supports the XC2000, XC3000, XC3100 and XC4000 versions from the Xilinx FPGA family

B 7 A case study- Viterbi Decoder

A Viterbi decoder, though not requiring any multiply operation, is however an example of a DSP algorithm The real time mathematical processing of the input

data stream makes it unsuitable for a DSP processor based design The main limiting factor in the DSP based design is the design and timing of the external SRAM Also, each Add/Subtract and Multiplex stage must be performed sequentially in the DSP

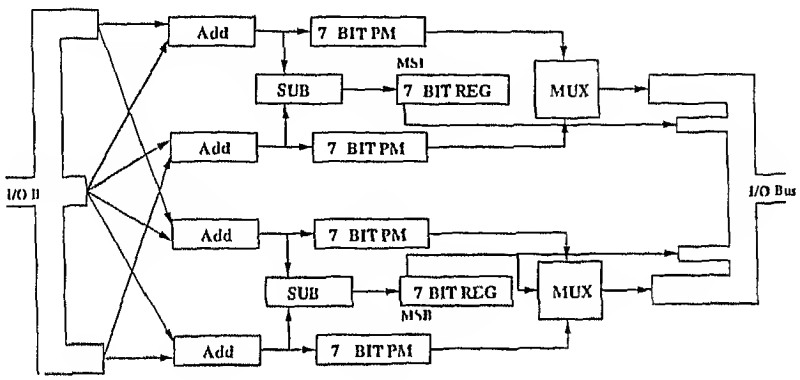


Figure B 3 Viterbi block diagram

The algorithm has been shown graphically in Fig B 3 This algorithm is well suited for implementation based on FPGAs The ability of FPGAs to process parallel data paths enables the implementation of the parallel structures for the four ADD/SUB blocks in the first stage and the two SUB blocks in the second stage The two MUX blocks take advantage of the ability to register and to hold data until needed, with no additional clock cycles

However, higher constraint lengths call for partitioning of the design before mapping it to the FPGAs For our design, with constraint length K=9, we had to partition our ACS unit, and the memory banks had to be mapped separately This can cause some unforseen delays which need to be taken into account while simulating the Verilog code

References

[1] John K Hinderling et al CDMA Mobile Station ASIC *IEEE JSSC*, vol 28 no 3, March 1993

[2] Ryuji Kohno et al ,"Spread Spectrum Access Methods For Wireless Communications," *IEEE Communications Mag* Jan 1995

[3] Lee *Mobile Cellular Telecommunications* 2nd Ed ,McGraw Hill

[4] G R Cooper et al ,"A Spread Spectrum Technique For High Capacity Mobile Communications," *IEEE Trans On Vehicular Tech* Vol 27 Nov 1978

[5] D Verhulst et al ,"Slow frequency Hopping Multiple Access For Digital Cellular Telephone," *IEEE JSAC,SAC 2* no 4 July 1984

[6] N Livnch et al ,'Frequency Hopping CDMA For Cellular Radio,"*Proc*, *ICS ,Herzilya,Israel*, Dec 1991

[7] R C Dixon,*Spread Spectrum Systems With Commercial Applications* New York Wiley Interscience,1994

[8] Klein S Gilhousen, "On the capacity of a cellular CDMA system,"*IEEE Tran Veh Tech* , vol 40, no 2, May 1990

[9] Klein S Gilhosen et al "Increased Capacity Using CDMA For Mobile Satellite Communication," *IEEE J Sel Areas Of Comm* vol 8, no 4, May 1990

[10] Gary D Boudreau et al "A Comparison Of Trellis Coded Versus Convolutionally Coded Spread Spectrum Multiple Access Systems *IEEE J Sel Areas In Comm* , vol8, no 4, May 1990

[11] G Ungerboeck "Trellis Coded Modulation With Redundant Signal Sets Part 1 Introduction" *IEEE Comm Mag* , vol 25 no 2, Feb 1987

[12] Riccardo De Gaudenzi et al , 'A Performance Comparison Of Orthogonal Code Division Multiple Access Techniques For Mobile Satellite Communications,' *IEEE J Sel Areas In comm* , vol 13 no 2 Feb 1995

[13] Walsh

[14] R Gold ,"Optimal Binary Sequences For Spread Spectrum Multiplexing" *IEEE Tran On Inf Theory* , Oct 1967

[15] Jerold A Heller and Irwin M Jacobs,"Viterbi decoding for satellite and space communication," *IEEE Trans Commun Technol* , vol COM 19 Oct 1971

[16] G David Forney,Jr , 'The Viterbi algorithm ' *Proc IEEE*, vol 61 Mar 1973

[17] Andries P Hekstra ,"An alternative to metric rescaling in viterbi decoders," *IEEE Tran On Commun* vol 37, Nov 1989

[18] Charles M Rader ,"Memory management in a viterbi decoder" *IEEE Tran On Commun* COM-29, Sept 1981

[19] Marc Biver et al ,"In place updating of path metrics in viterbi decoders," *IEEE JSSC* vol 24, Aug 1989

[20] Q1650 Qualcomm Data sheet 1987

[21] Qualcom,"K=7 rate 1/2 single chip Viterbi decoder technical data sheet'

[22] St anford Telecommun ,"20 Mbps convolutional encoder decoder STEL 2020

[23] J St ahl and H Meyr "A bit serial Viterbi decoder chip for the Mbps range *Proc IEEE 1987 Custom Int Circuits Conf pp 551 554*

[24] Shung, Lin et al "Area Efficient Architecture for the Viterbi Algorithm - Part I," *IEEE Tran on Commn* vol 41 No 5, May 1993

[25] H J Siegel, *Interconnection Networks for Large Scale parallel Processing Theory and Case Studies* Lexington MA Lexington Books 1985

[26] J D Ullman *Computational Aspects of VLSI* Computer Science Press 1984

[27] D J Coggins et al ,"A comparison of path memory techniques for VLSI viterbi decoders," *VLSI 89* pp 379 388

[28] J Sparso et al ,"An Area Efficient Topolgy for VLSI Implementation of Viterbi Decoders and Other Shuffle Exchange Type Structures" ,*IEEE J Solid State Circuits* vol 26, no 2 Feb 1991

[29] P Gulak and E Shwedyk,"VLSI structures for Viterbi receivers Part I," *IEEE J on Sel Area In Commun* vol SAC 4, pp 142 154 Jan 1986

[30] P Gulak and T Kailath,"Locally connected VLSI structures for the Viterbi algorithm ," *IEEE J on Sel Area In Commun* , vol SAC 6, Jan 1988

[31] P J Black and T H Y Meng,"A 140 Mb/s, 32 state, Radix 4 Viterbi decoder," *IEEE J Solid State Circuits*, vol 27, no 12, Dec 1992

[32] C H Papadimitriou and K Steiglitz *Combinatorial Optimization Algorithms and Complexity* PHI, 1997

[33] Michael A Bree et al , "A modular bit serial architecture for large constraint length Viterbi decoding," *IEEE J Solid State Circuits*, vol 27, no 2, Feb 1992

[34] Normand J P Prenette et al " Implementaton of a Viterbi processor for a digital communictions system with a time dispersive channel ' *IEEE J Sel Areas In Commun* , vol SAC 4 no 1 Jan 1986

[35] Cadence, *Verilog XL reference*

[36] Chandrakasan et al , "Low Power chipset for portable multimedia I/0 terminal, ' *IEEE J Solid State Circuits* vol 29 no 12 Dec 1994

Application Specific Digital Signal Processing performance ,' *Xilinx Inc*

[37] Xilinx *The programmable Logic Data Book*

[38] Xilinx *XBLOX manual*

[39] Gregory Ray Goslin ,"A guide to using Field Programmable Gate Arrays (FPGAs) for Application Specific Digital Signal Processing Performance ," *Xilinx Inc*

EE-1998-M-PAN-DES